

Capstone Whitepaper - Auction Price Prediction & Image Features

Bingying Liu, Yifei Wang

Abstract - We propose a multi-layer price prediction model that concatenates a pretrained Convolutional Neural Network (CNN) image model, preprocessed tabular data, text representations and other features to predict eventual sales price. We benchmark performance of different hyperparameters and optimize the final price prediction with mean absolute error \$2541.85 (19.06%). We conduct quantitative feature importance analysis of all the interpretable features by giving a importance ranking. We also implement two qualitative network attention algorithms on images (image attention) by color coding important areas or pixels. These interpretations could help our clients understand how important each feature is to the final price. People with professional knowledge could potentially extract consistent recommendation from the image attentions and optimize the background of images.

1. Project Background and Goals

Background about Purple Wave

Our partner is Purple Wave, the second largest online heavy machinery auction firm. Its auction acts like “Ebay”, but focuses on agricultural and construction equipment. Purple Wave has huge volumes of past transaction records and equipment images, in which they find values for learning what aspects of the equipment conditions are affecting the final auction most. Ideally we could build a dynamic price prediction model that jointly considers both the tabular information (make, year, mileage, etc.) and the interpretable and adjustable features extracted from images. In this way, Purple Wave could maximize their profits, by tuning the listed information of equipment (e.g. picking a cover image that looks great) and maximizing the expected final price.

We use images of skid steer, a small piece of construction equipment primarily used for digging, and auction information in our project, because these machines are auctioned most frequently. We can utilize the great amount of skid steer data to build the models and easily adapt them to other equipment.

Importance of Interpretable Price Estimation

Price estimation is crucial to the auction business, because buyer’s premium is a main source of their revenue. Buyer’s premium is usually a set percentage of the final price, paid by the buyer to the auction company as a service fee. The more expensive the sold item, the more money

the auctioneer will make from charging the buyer's premium. Accurately estimating the selling price in advance could help auction companies optimize their resources to arrange the auction so that similar items won't be sold at the same time. It could also help them advertise accordingly and maximize their revenue.

In addition to price prediction, feature interpretability is also crucial to auction business. Knowing which features are important to the final price, auction companies could extract patterns and formulate an optimized way to improve customer satisfaction. For example, professional photographers of Purple Wave could give an image more visual appeals, or promote/advertise items by emphasizing certain features. All these approaches could potentially increase customer's satisfaction by achieving a higher auction price.

Overview of Findings

In this project, we experiment on and compare different ways of generating image features. Machine learning methods perform poorly on generating image features without annotations, while a proper way of image processing could extract meaningful and significant features. Another great way to gather image features is using Amazon Mechanical Turk (MTurk). In terms of generating image features, we suggest using MTurk or other human annotations to gather meaningful features.

As for the price prediction model, we use a neural network to build a flexible end2end price prediction model taking skid steer images, preprocessed text comments and all other vehicle features as inputs. We experimented on different combinations of and hyperparameters the model structures and finally reached a mean absolute error of \$2541.85 (mean absolute percentage error of 19.06%) for predicting unseen data.

We also conduct quantitative feature importance analysis on all numerical vehicle features and qualitative analysis on raw images. We find that the vehicle body color and cleanliness play an important role in price prediction. Thus, we suggest cleaning the vehicle before taking photos.

2. Data Summary

We have two data sources provided by Purple Wave: **6197 skid steer images** and a **spreadsheet** on predictors and target selling price of 8172 skid steers. Because bidded items are usually pulled off from websites within several days, not all images could be retrieved, which results in a mismatch between the number of images and that of tabular information. However, all except one image could be matched with tabular information using a combination of item id and source. Both images and tabular data were collected by Purple Wave from four different sources: rbauction, ironplanet, purple wave and bigiron.

Key usage of tabular data is summarized below in *Table 1*.

Variables for modelling	Variables (duplicated/unused)	Variables to identify images
Year	Item Id	Source
Make	Product Name	Item#
Hours final	Year of sold date	
Winning Bid	Month of sold date	
Bucket	Date of sold date	
Engine		
Tires		
Transmission		
Age at sale		
Details remaining (Remaining details from listing less (bucket, engine, tires, trans))		

Table 1. Summary of variables in tabular data

Data Preprocessing

After exploring the raw data, we found that certain preprocessing procedures are necessary before feeding them to a machine learning model. Our preprocessing pipeline includes removing illegal rows (defined below), imputing missing values, transforming numerical values, normalizing numerical values, text cleaning, text modeling and train-validation splitting. Details and explanations for those steps may be found in the Appendix 1: Exploratory Data Analysis.

- Removing illegal rows

Illegal rows are rows with duplicate id or without matched images. We remove all the rows with duplicate “Item Id” otherwise we are not able to distinguish between them. Then we remove rows without matched images. We also found some corrupted image files so we also removed the corresponding rows.

- Imputing missing values

We found missing values in “Hours final” and “Age at sale”. For each of them, we impute these missing values with the corresponding median, and add a new binary variable indicating whether the original value is missing or not.

- Transforming numerical values

Some variables, such as “Winning bid” and “Hours final”, have great variance, which is generally not good for machine learning models. Thus, we transform these variables into log-scale.

- Normalizing numerical values

All these variables come from different ranges and scales. Some range from 0 - 42 (Age at sale), while others range from 5000 to 70000 (Winning bid in US Dollars). These will also cause difficulty in training an accurate model, so we normalize them to a standard scale.

- Categorical variable cleaning and merging

Variable “Make” is cleaned by inspecting and merging similar-looking brands together into one brand (for instance, treat “bob-cat” as “bobcat”) and combining brands with less than 20 rows of data together into a single category “Others”, which could enable “Make” to be less imbalanced.

- Text cleaning

Text that inputs into VADER’s sentiment analyzer are comments that exceed 100-character in length (from “Engine”, “Tires”, “Transmission” and “Bucket”) since we assume those texts are professional evaluations instead of machine specs. We use compound score as the output from VADER, which is a score from -1 (most extreme negative) to 1 (most extreme positive). We manually assign a compound score of 0 to comments that are less than 100 characters, as we assume shorter texts in the majority of cases are a combination of machine specs and its serial number, which has a neutral sentiment.

- Text modeling

We train different text embedding on the ‘details_remaining’ column based on the source of the data (rbauction, bigiron, PW and ironplanet) since the context looks very different based on source. Texts are decapitalized, punctuation removed and stemmed before inputting into fasttext to generate word embeddings. Sentence embeddings are calculated by summing word embedding in each sentence and average according to number of words.

- Train-validation splitting

Finally we split the whole dataset into a training set and a validation set. We set the portion to be 70% for training and 30% for validation. We also freeze the splitting to make sure that all the models will learn from the same training data and be tested on the same validation data.

From exploratory analysis (*Appendix 1*), we have predictors available directly from tabular data which has relationship with the response variable: winning_bid :

- Hours_final
- Age_at_sale

- Make
- Month of sold date

The variables above need minimum amount of preprocessing and have strong predictability power as Purple Wave is currently using for their own model. However, there are more factors that contribute to the attractiveness of a skid steer than structured numerical data, such as image and text descriptions on the web. In this section, we hope to use well-established rules from research to extract useful features from unstructured data, as well as crowdsource human knowledge to annotate image features.

Feature Engineering

On the Purple Wave website, each equipment has an enlarged cover image showing general looks of the equipment from the front view. This cover image would have greater influence on the buyer's first feelings than other detailed images. Buyers will not even have a chance to see the detailed images if the cover image is not attractive enough. Thus, one of the main goals of our project is to determine what aspects of the cover image could influence product attractiveness and how important those relevant aspects are regarding eventual sales price prediction. In addition, suggestions could be made to Purple Wave on how to improve cover image to attract more traffic.

1) Image Colorfulness

An important external factor that influences customer's perception of a product is color saturation, or "happiness" of the image. For example, drawing on social cognition theory and experimental studies, Schnurr^[1] concluded that customers perceive products as more attractive when they are put in attractive contexts.

'Happiness' of the image in our case, can be broken down into subcategories, for instance weather of the image-taking day (i.e. sunny,cloudy, snowy,etc), background of the image (i.e. inside garage, outside under blue sky, etc) and quality of photos taken (i.e. overexposed, underexposed,etc). In addition, shade variants of equipment's color could also be important.

In the paper "Measuring colourfulness in natural images", Haslera and Susstrunk^[2] asked 20 participants to rate images using 7 categories of colorfulness. This survey was conducted on 84 images and a simple metric was proposed that correlated to 95.3% of the experiment data. The metric uses opponent color space representation, which is the difference between red and green as well as yellow and blue channels (*Figure 1*).

$$\begin{aligned}
Rg &= R - G \\
Yb &= (1/2) (R + G) - B \\
\sigma_{rgyb} &= \sqrt{\sigma^2_{rg} + \sigma^2_{yb}} \\
\mu_{rgyb} &= \sqrt{\mu^2_{rg} + \mu^2_{yb}} \\
C &= \sigma_{rgyb} + 0.3 * \mu_{rgyb}
\end{aligned}$$

Figure 1. Colorfulness Metric

This is a general metric which essentially includes almost all subcategories of ‘Happiness’ mentioned above. As we can see in *Figure 2.a,b*, we sampled 50 images, calculated their colorfulness scores and sorted them in descending and ascending order.



a. 10 most colorful images



b. 10 least colorful images

Figure 2. Samples of image colorfulness

In the most colorful images, skid steers usually have a background of blue skies, pastures or the color of “yellow” is quite bright. Whereas in the least colorful images, skid steers’ backgrounds involve snow and cloudy weather or the color of skid steers aren’t quite bright. In the exploratory data analysis (EDA) section (*Appendix 1*), we can see the relationship between price of skid steers and colorfulness scores. Although there is not a strong positive correlation between final price and colorfulness in EDA because of other dependent variables, we will use the nested-f test later in linear regression to prove its significance towards price prediction.

Although colorfulness score is simple and effective, we still would like to break down this score into more interpretable subcategories, which will be implemented in the annotation and transfer learning section.

2) Text Sentiment

Apart from the images, text descriptions of the equipment are also alongside on the auction page. Purple Wave has scraped the text data and stored them inside different columns in the spreadsheet. Although text might not have as strong a first impression as the cover image, we still believe that text could convey extra information that couldn't be expressed by images. Buyers could be benefited by knowing machine specs, professionals' evaluation and etc.

As we explored the corresponding columns and found that the seemingly "categorical" variables, specifically 'Engine', 'Tires', 'Transmission' and 'Bucket', actually contain more than hundreds of unique categories (out of 6k+ data), treating them as "categorical" instead of unstructured text will dilute the effectiveness of the variable itself (Appendix 1: Text Data). Also, since the tabular dataset is collected from four different sources and some of the comments contain more than 100 characters, it's technically easier and reasonable to extract sentiments from text. Sentiment in price prediction's context refers to identifying positive or negative sentiment within text that could potentially drive sales price.

We use VADER (Valence Aware Dictionary and Sentiment Reasoner)^[3], a lexicon and rule-based sentiment analysis dictionary that has been shown to have good performance on product review dataset^[3]. VADER's advantage is that it takes into account preceding trigrams, which means negation and conjunction situations can be dealt with properly. We specifically use the compound score, which is a metric that calculates the sum of all the lexicon ratings to extract sentiment from 'Engine', 'Tires', 'Transmission' and 'Bucket' columns. In the feature importance section, we'll see how these sentiment ratings could impact the sales price. One of the disadvantages of VADER is that since each word's sentiment is hard-coded, a word like "charged" has a neutral meaning in our dataset but is given a negative sentiment in the dictionary. We manually modified several words' lexicon rating, but this dictionary needs to be further explored and maintained once new data comes in.

```
engine access door latch area damaged, engine knocks, engine shroud damaged, new holland 450nc 5.0l four cylinder diesel engine
{'neg': 0.254, 'neu': 0.746, 'pos': 0.0, 'compound': -0.7003}
81 hp, approximately one hour on replaced engine, new holland 445m2 four cylinder diesel engine, recently replaced engine
{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}
70 hp, john deere 5030ht014 3.0l five cylinder turbo diesel engine, one engine cover side panel missing
{'neg': 0.121, 'neu': 0.879, 'pos': 0.0, 'compound': -0.296}
engine compartment fire damage, john deere pe5030 four cylinder turbo diesel engine, lift arm cylinder pins removed, but included
{'neg': 0.183, 'neu': 0.817, 'pos': 0.0, 'compound': -0.4215}
```

Figure 3. Output scores from VADER using 'Engine' column

Let's walk through an example of the output from VADER. For the first sentence, VADER identifies "damaged" as a negative word while it doesn't mark "engine knocks" as negative. The

reason is that “knock” itself is a neural word; only by treating “engine knocks” as a phrase can it have specific meaning. However, for a lexicon-based library, it has its own limitations. Similarly, in the second sentence, “replaced” here could have positive sentiments. However, “replaced” could actually have different sentiments under different circumstances. Again, VADER is unable to detect the context of a word by its library nature.

3. Image Annotation (MTurk)

Since it’s generally hard to extract interpretable image features by image processing techniques alone and colorfulness scores incorporate so many aspects of the pictures, we resorted to a crowdsourcing platform to gather human annotations and break down the score.

Amazon Mechanical Turk is a popular crowdsource platform for computer vision people to get image labels, bounding boxes of targets, etc. It’s a cheap and fast way to collect data. Improved upon the first data collection experience, we worked with Purple Wave to design 13 questions (Appendix 4a: Questionnaire Design) that could potentially extract as much information from the images as possible. Questions involve yes-no questions to determine background, quantitative annotations such as rust extent, brightness of color, dirt level and etc. We conducted a sample trial by sampling 100 images from an image set and getting 3 workers to work on each image. The sample trial cost \$83.94 in total. We designed the MTurk interface like below (*Figure 4*), and also provided detailed guidance for workers to understand (Appendix 4b: Detailed instructions for workers to label bucket rust extent) what different extent of brightness/rust/dirt level looks like by providing sample images.



1. Does this equipment's bucket contain rust?

☐ Yes ☐ No ☐ This equipment has no bucket.

2. Use a scale of 1 to 10 to describe how much rust the body of equipment has (please only take in account the body, not the bucket).

1 means 'barely any rust' and 10 means 'rust all over the body'.

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8 ☐ 9 ☐ 10

Submit

Figure 4: MTurk question interface

To prevent illegal selections, we designed the survey using radio buttons and javascript hide elements, which significantly reduced the number of illegal answers. We used majority votes to preprocess the data: if 2 workers have the same labeling, we pick them as the truth one. We

have 96 rows of legal annotations since one of the 13 questions designed is a 3-choice question, and we happened to have annotators pick three different answers for 4 images. Therefore, we excluded the controversial images, which left us with 96 image annotations. We also explored the numerical variables' annotation quality in Appendix 4c: MTURK Annotation Quality. In the later section, we explore the effect of adding mturk data in the model to investigate if features in images are important.

4. Multi-layer Price Prediction Models

1) Baseline Price Prediction Model & Significance Tests

We started with using multiple linear regression (MLR) as our baseline model as well as a tool for checking significance of added features. Because firstly, it is highly interpretable and secondly significance tests could be implemented in the regression setting. In addition, a H2O random forest is fine-tuned to improve the prediction accuracy. As we mentioned in Appendix 1: Numerical Data, we found a log relationship between predicted variables and different predictors, so we perform a log transformation on Hours_Final, Winning_Bid and Age_at_Sale. Categorical variables are one-hot encoded in regression models.

We conducted a series of model comparisons in Table 2. Note that models colored in purple were trained and tested with the dataset of sample size 6167 (tabular information alone); Models colored in yellow were trained and tested with the dataset of sample size 96 (joined with MTURK).

2) Tabular data + extracted features (sample size: 6167):

By adding features one at a time, we can check the significance of added predictors and observe the improvements of metrics in Table 2 (increase in R-squared and decrease in MAE, concepts explained in detail in the caption). By performing the nested-f test on each new feature, we found that all features except tire and transmission sentiment are significant. In addition, with all features included, we have a multiple R-squared of 0.436 and MAE of \$3926 for this baseline regression model.

Next, H2O random forest (h2o_drf) is implemented to increase the predictability power. One of random forest (rf)'s advantages over linear regression is that rf is not affected by multicollinearity between features. In addition, rf takes in account nonlinear transformations and considers interactions without specifying them due to its tree-based nature. H2O drf is preferred in this scenario because categorical variables are not one-hot encoded like in many other packages [4]. One-hot encoding degrades the performance of tree-based models since it creates many independent binary variables, which during splitting are disadvantaged over continuous variables.

After fine-tuning (grid search over hyperparameters), H2O random forest model achieved a MAE of \$3391, which outperforms the MAE of baseline regression model (\$3902) [hyperparameters: ntrees=100, max_depth=10, sample_rate = 0.8, mtries=3]. This shows that during this stage of modelling, our price prediction error is around \$3391 on average where the winning bid ranges from \$5000 to \$70,000.

3) Tabular data + extracted features + MTURK annotations (sample size: 96)

Since we've collected 96 rows of annotations for images, we also joined the original data with the newly collected data to see the performance of regression models. In Table 2, we can see that using multiple linear regression with 96 rows of data, we could achieve a multiple R-square of 0.802 using all data, which is a significant increase in R-square (0.431) by using tabular data alone. However, due to the small sample size and substantial amount of features, there is an overfitting problem in the models, resulting in unstable MAE. This instability is due to the small sample size relative to the number of predictors as well as regression's incapability of dealing categorical variables with many levels. We can have as many as 49 predictors when implementing one-hot encoding in regression while having 96 data points. Therefore, we'll try to use a more powerful model to perform fine tuning. Nevertheless, we also suggest Purple Wave to collect more annotations in the future to test mturk features' significance with a higher confidence level.

We experimented with H2O random forest for this small dataset. After fine-tuning, we achieved a MAE of \$2978, which shows the effectiveness of annotations (additional features) in price prediction.

Model	Parameters	Sample Size	R^2	MAE	Nested-f test (significance)
MLR	Hours_final, age_at_sale	6167	0.392	4076	*
MLR	Hours_final, age_at_sale, parts_sentiment	6167	0.394	4062	Engine & Bucket Sentiment
MLR	Hours_final, age_at_sale, colorfulness_score	6167	0.396	4047	*
MLR	Hours_final, age_at_sale, month_of_sold_date	6167	0.401	4066	*
MLR	Hours_final, age_at_sale, make	6167	0.430	3926	*
MLR	Hours_final, age_at_sale, make , month_of_sold_date , colorfulness_score , parts_sentiment	6167	0.436	3902	*

H2O_DRF	Hours_final, age_at_sale, make, month_of_sold_date, colorfulness_score, parts_sentiment	6167	0.489 (MSE)	3391	
MLR	Hours_final, age_at_sale	96	0.431	5384	
MLR	Hours_final, age_at_sale, make	96	0.570	4998	
MLR	Hours_final, age_at_sale, make, mturk	96	0.633	Unstable Because of Overfitting (Linear regression is not good at dealing with one-hot encoded categorical variables. In addition, the total number of predictors adds up to 49 when one-hot encoded and 19 when label encoded. 96 is too small a sample size to test significance with high confidence level .)	
MLR	Hours_final, age_at_sale, make, mturk, parts_sentiment, colorfulness	96	0.676		
MLR	Hours_final, age_at_sale, make, mturk, parts_sentiment, colorfulness, month_sold	96	0.802		
H2O_DRF	Hours_final, age_at_sale, make, mturk, parts_sentiment, colorfulness, month_sold	96	0.850 (MSE)	2978	

Table 2. Model Comparison

- Models colored in purple were trained and tested with dataset of sample size 6167 (tabular information alone); Models colored in yellow were trained and tested with dataset of sample size 96 (joined with MTURK)
- R-squared, denoted by $1 - \text{RSS}/\text{TSS}$ where RSS is the residual sum of squares and TSS is the total sum of squares, represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model^[5]
- Mean absolute error (MAE) denotes the average of all absolute errors.
- Nested-f test is used to test if multiple coefficients equal zero or not. “*” in the plot above represents that the variables added are significant; ‘engine & bucket’ means that for machine specs’ sentiment scores, only sentiments of engine and bucket are significant.
- Metrics above are all calculated using the test set (hold-out set).

4) Sentence Embedding for Text Data

In order to make use of ‘details_remaining’ column, which is the rest of information that doesn’t belong to “Engine”, “Tires”, “Transmission” and “Bucket”, we decided to generate sentence embeddings for text and use a regression in pytorch framework to test its effectiveness.. The reason we do not use sentiment analysis again is that texts coming from rbauction (one of the four sources), which counts for 60% of all data, are concatenations of abbreviations of machine specs. Text from other sources are long comments given by heavy equipment professionals. We trained corpus-specific word embeddings using fasttext and fed sentence embeddings as input vectors. For sentence embedding alone as input features, we got an R-square of 0.42,

while tabular feature alone has R-square of 0.41. The two combined achieved an R-square of 0.60. The explained proportion from R-square tells that sentence embedding can improve price prediction but still interpretability of sentence embedding is a problem yet to be solved.

5) Transfer Learning for Images

We show in the previous section that it is difficult to extract interpretable image features through machine learning without the guidance of human knowledge. Thus, we decided to use images to predict price directly. Specifically, we decide to use transfer learning to solve this problem. Transfer learning means fine-tuning the weights of a pre-trained model to our task of price prediction. We compare the performance of different pre-trained models (see details in Appendix) and decide to use ResNet152 as our image model because it has the best balance between price prediction performance and training time complexity.

The pretrained ResNet152 model is originally used in classification tasks, predicting the class of the images (e.g. whether the image has a cat or dog). Our task, however, is a regression problem, predicting the final auction price. To address the difference, we split the original ResNet152 model into two different parts. The first part contains all the convolutional layers, pooling layers and activation layers and it takes an image as an input and outputs an array of uninterpretable features. We keep the structure of this part unchanged. The second part contains all the fully-connected (dense) layers and it takes the output array of the first part and produces a final output for this model. We modify the second part to produce a price rather than the class. Finally we concatenate these two parts back together and train these two parts at the same time.

6) Training the Multi-layer Price Prediction Model

Next, we need to implement a way to incorporate the baseline features, text information and the uninterpretable image features together in predicting auction price. As mentioned before, all these information are array values, so we concatenate them together into a long array. Then we extend the fully-connected layers so that it could take this informative long array as an input and output the price we need. (Figure 5)

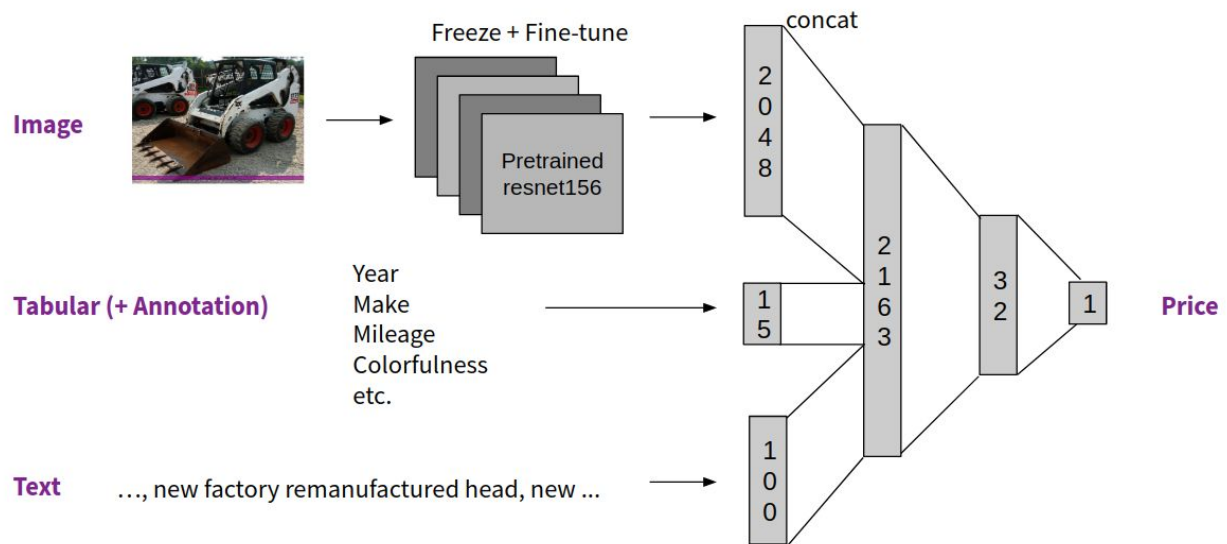


Figure 5 : Structure of the multi-layer price prediction model.

We also conduct performance benchmarks on different settings of hyperparameters (more details in Appendix). Hyperparameters are parameters that affect prediction performances and are manually chosen by humans but not automatically learned by machine. We finally choose these hyperparameters set in our final model displayed in Table 3.

Hyperparameter	Values
Pretrained image model	ResNet152
Fine-tuning CBs	CB3+
Optimizer	Stochastic Gradient Descent
Loss function	Mean Square Error (MSE)
Fully-connected structure	(2308, 32, 1)
Activation functions in fully-connected layers	Hyperbolic tangent
Training epochs	200
Starting learning rate	0.001
Scheduler	Reduce learning rate on plateau

Table 3: Hyperparameters combination for the final model. Each training epoch loops through the whole training dataset. Learning rate shows how fast the model is learning from training data. The scheduler we choose will reduce learning rate when performance on validation set stops improve so to improve the overall performance. Meaning of other hyperparameters could be found in Appendix 3

5. Model Performance Evaluations

1) Feature Importance

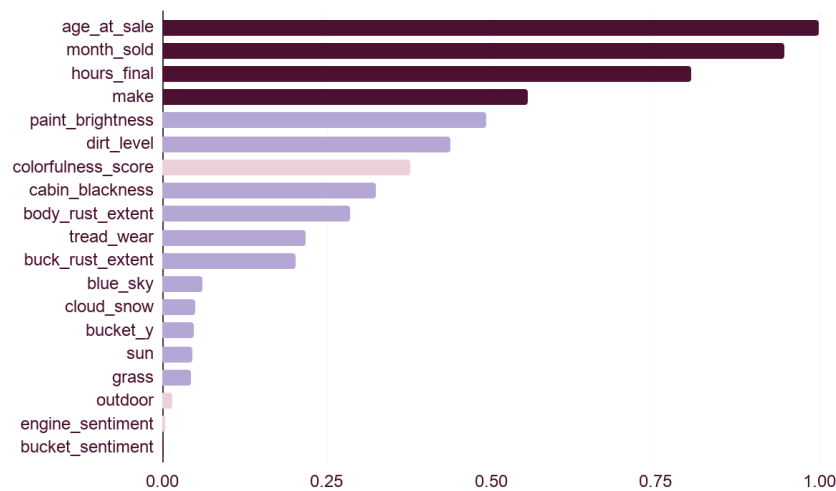


Figure 6: Feature importance generated by Random Forest Regressor

Dark purple: Original tabular data; Light purple: MTURK data; Light pink: Feature extracted

From the random forest regressor's feature importance plot, we could see that `age_at_sale` is the most important predictor, followed by `month_sold`, `hours_final`, `make` and `paint_brightness` (top 5 predictors). In Appendix 5: Categorical Feature Importance Breakdown, importance of subcategories of categorical variables are shown. For instance, holding other variables constant, equipment sold in October tends to have a higher selling price than equipment sold in other months. One of the reasons is that October is a peak season for sales. Also, Caterpillar is a good/much valued brand that has equipment's sales price higher than other brands' equipments' sales price. It also might be due to the imbalance of categories in this dataset as shown in the exploratory analysis section.

In general, text information has less predictive power than image features, than primary predictors such as `age_at_sale` (text < image < primary predictors). But we can't deny the importance of features shown in Figure 6 and Table 2. Therefore, we suggest Purple Wave to start collecting relevant data (mturk survey questions) in the future to train a better model.

2) Prediction Accuracy

We compare our predictive prices with the ground truth price on the validation set. We use 3 metrics, loss function (MSE in log scale), Mean Absolute Error (MAE) in US Dollars and Mean Absolute Percentage Error (MAPE) in percentage scale, to measure our prediction performance. The results are shown in the table.

Variables used	Loss (log-scale MSE)	MAE (original scale)	MAPE
Baseline	/	4076	/
+ Image (colorfulness score)	0.0331	2618.11	19.34 %
+ Text embeddings	0.0332	2541.85	19.06 %
+ Text sentiments	0.0323	2598.10	19.07 %

Table 4: Benchmarking the same model with different parameters used. Baseline contains all the variables in the original tabular. Model in each row utilizes all the variables shown above the current row.

	Loss (log-scale MSE)	MAE (original scale)	MAPE
Training Set	0.0236	2243.50	15.83 %
Validation Set	0.0317	2541.85	19.06 %

Table 5: Prediction accuracy on training and validation set.

All metrics have similar values between training and validation. This indicates that we are not suffering from overfitting, and the model has consistent generalization performance. The MAE shows that our price prediction error is around \$2600 (19 %) on average.

3) Visualizations

In addition to price prediction, we also want our model to have interpretability. We already gave interpretation on the tabular features showing which contribute more to the final price. As for images, it is difficult to derive the same quantitative features importance rank. Instead we want to qualitatively visualize the attention of our network. The network attention are areas that have

great impact on producing the final output (price). In this way, we could show which areas are driving the model towards a higher/lower final price.

The first algorithm we implement is called Class Activation Maps (CAM). It is originally used in classification problems, visualizing which areas are driving the network in predicting the given class. Our network, however, is a regression model. We modify this algorithm to fit our purpose. Some sample results are shown in the Figure 7 below.

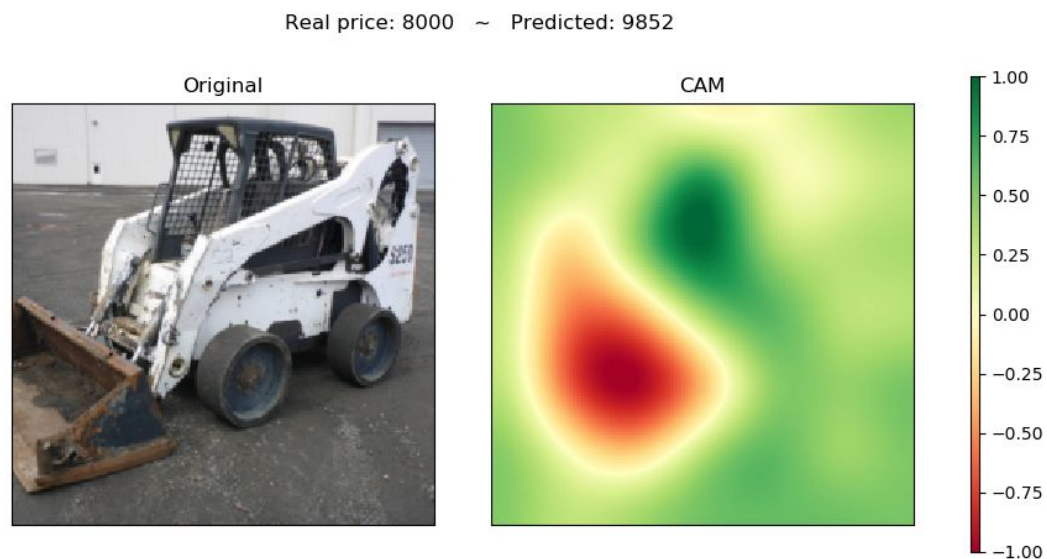


Figure 7: A sample of CAM visualization. The right is the CAM visualization, showing the network attention on the left original image. The greener the areas, the more positive attention the network is giving to. The redder the areas, the more negative attention the network is giving to. The more positive/negative attention the area has, the more it drives up/down the final price. Our model favors a black cabin frame, clean body color, clean background and dislikes the rusty bucket and conjunction parts.

The second algorithm we implement is called Guided Saliency Maps. Similar to CAM, Guided Saliency Map is also originally used in classification problems. We modify the algorithm to fit our regression model. It shows the pixel-wise gradients of the ground truth price with respect to the input images. A sample is shown in Figure 8 below.

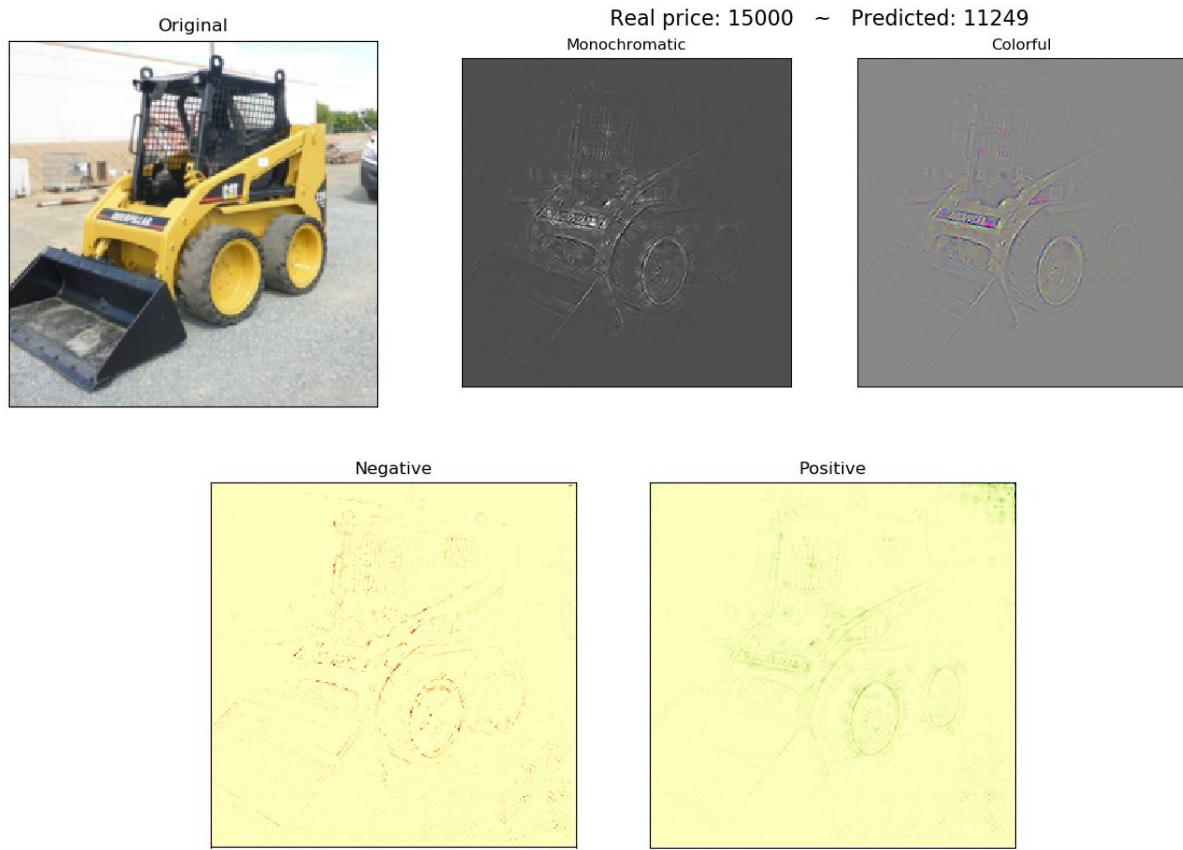


Figure 8: A sample of Guided Saliency Map visualization. Counting clockwise from the top left image, the first shows the original image. The second shows a monochromatic gradient, which is taking the pixel-wise max gradient values from the gradients in RGB channels in the third colorful image. The fourth/fifth image only keeps the negative/positive gradients from the second image. The brighter or more colorful the pixels, the more impact of the pixels on the output price.

Guided Saliency Map is calculated pixel-wise with the same shape of the input images. It contains detailed information on network attention, but the visualization might be too detailed and lose focus. CAM, on the other hand, first separates the input images into 7 by 7 small square areas and then calculates the attention on each small area. This makes the output visualizations look like patches of blue and red color. By putting these two techniques together, we could better understand where the network is paying attention to.

Neural networks are always referred to as a black box, because they produce an output from a given input without an explanation. Thus visualizing neural networks is always a hard research question. We use attention to visualize the important part of the images as a way to achieve the feature importance in the tabular data. It is a qualitative approach but not a quantitative

approach. Thus, these visualizations require further professional investigation to see if consistent patterns exist within.

Finally, we incorporated all the visualizations in an app deployed on heroku

6. Discussion

We believe using MTurk, or any other human annotation approach, is the easiest and most cost-effective way to gather interpretable (thus actionable) and significant features. A well-designed MTurk questionnaire could make as high as 96% of the answers useful, with the cost of less than 0.9 USD per image. In addition, people with professional knowledge about pre-owned skid steer could then justify those answers from MTurk by sampling. Thus it results in both authentic and cost-effective feature labels.

We see a great performance improvement after adding raw images in our price prediction model, but this effect diminishes when adding some other new features (comment sentiment and embeddings). One hypothesis is that the network tends to weigh a lot on the images (2048 out of 2163 features are for images). A possible solution for this is to add another fully-connected layer to the 2048 image features before the concatenation (such as reducing from 2048 to 512). This will increase the model complexity and thus make it might need more data to train on.

For the analysis on feature importance, we find that vehicle paint and cleanliness have significantly great impact on predicting price. However, we lack a constant interpretation on images with the two proposed attention methods. Researchers have been using these methods on classification problems but not regression problems like price prediction. This migration of problem settings could have a huge impact on the performance of these two algorithms. Thus, we suggest looking at these visualizations and recording their interpretation one by one. Then we could compare (or count) the frequency for each type of interpretation, to finally discover a constant understanding of the model's preference.

7. Citations

[1]. Benedikt Schnurr, *The effect of context attractiveness on product attractiveness and product quality: the moderating role of product familiarity*, 09 August 2016

[2]. David Haslera and Sabine Susstrunk, *Measuring colourfulness in natural images*, 2003

[3] C.J.Hutto, Eric Gilbert, VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text, 2014

- [4]. Nick Dingwall, Chris Potts, Are categorical variables getting lost in your random forest, <https://roamanalytics.com/2016/10/28/are-categorical-variables-getting-lost-in-your-random-forests/>
- [5]. Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, An Introduction to Statistical Learning with Applications in R
- [6]. Julianne Alyson I. Diaz, Manuel I. Ligeralde Jr., John Anthony C. Jose, etc., *Rust Detection Using Image Processing via MATLAB*, 2017.
- [7]. Margarita R.Gamarra Acosta, Juan C.Vélez Díaz and Norelli Schettini Castro, *An innovative image-processing model for rust detection using Perlin Noise to simulate oxide textures*, 2014.
- [8]. Jason Yosinski, Jeff Clune, Yoshua Bengio, etc., *How transferable are features in deep neural networks?*, 2014.
- [9] Stéphane Lathuilière, Pablo Mesejo, Xavier Alameda-Pineda, Radu Horaud, A Comprehensive Analysis of Deep Regression, 2019

8. Appendix

1) Exploratory Data Analysis

We explored the relationships between the winning bid and predictors from tabular data. Missing values and data anomaly problems are dealt with in this process. The following is organized in the order of missing values, numerical, categorical and text data.

- **Missing Value in Tabular Data**

For various add-ons such as ‘bucket’, ‘engine’, ‘tires’, etc, they all have missing data. Their missingness is majorly due to different data sources’ ways of annotation. Some websites only write very specific comments in a case-by-case fashion; whereas the others put the category of add-ons in the column. Number of null values in variables are given in *Figure 9*.

Item Id	0
Source	0
item#	0
Product_Name	0
Year	429
Make	0
Model	0
Hours_Final	637
Winning_Bid	0
Bucket	1347
Engine	6196
Tires	5304
Transmission	7102
Age_at_sale	430
details remaining	19
url	0
Year_of_Sold_Date	1
Month of Sold Date	1
Day of Sold Date	1

Figure 9. Number of null values in variables

Bucket has a more complicated situation. Null value can either be caused by skid steer itself lacking this bucket or incomplete annotations. According to human observation, there are situations that although tabular data has a null value in 'Bucket', associated image has a bucket on it and vice versa. We'll talk about how to deal with the missing value in exploratory analysis: text and categorical variable section.

- Numerical Data

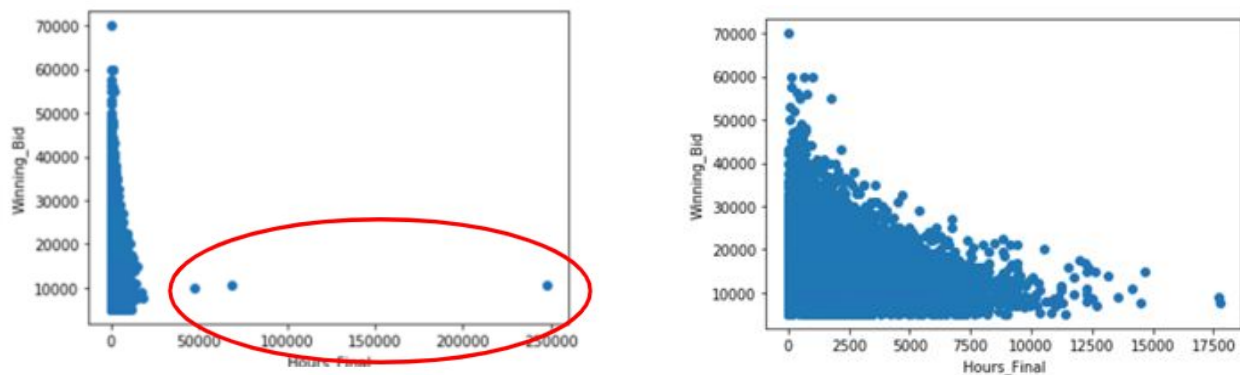


Figure 10. Exploratory Plots of Winning Bid vs. Hours Final

Plot on the left-hand side shows that there are 3 outliers with more than 40,000 hours listed in the meter. When we removed the 3 outliers, we have the plot on the right-hand side.

From Figure 10, we can see that there are outliers with more than 40,000 hours listed in the meter. Since the majority of prediction lies in the range of 0 to 40,000 hours, we chose to exclude the outliers and saw a trend of exponential/quadratic drop (variable needs to be transformed). We can see a similar relationship between winning bid and age of the skid steer in

Figure 11. In case of multicollinearity, we derived the correlation between numerical variables and found there is a correlation of 0.198 between Hours Final and Age at sale, which was acceptable.

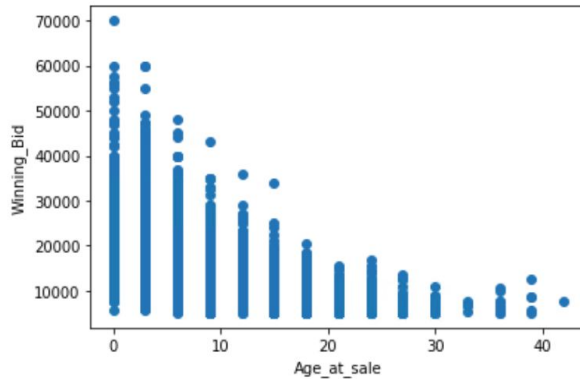


Figure 11. Winning Bid vs. Age at sale

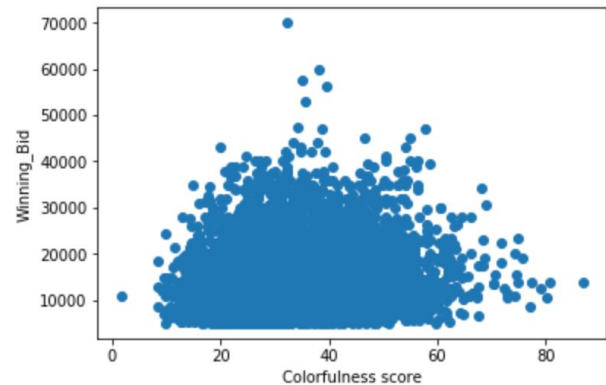


Figure 12. Winning Bid vs. Colorfulness Score

We then explored the relationship between the winning bid and the colorfulness score. From *Figure 12*, we can't see a significant trend between dependent and independent variables due to the fact that this is a marginal plot which isn't conditioned on other predictors. We'll figure out the influence of colorfulness to price when putting it into a baseline regression model.

- **Categorical Data**

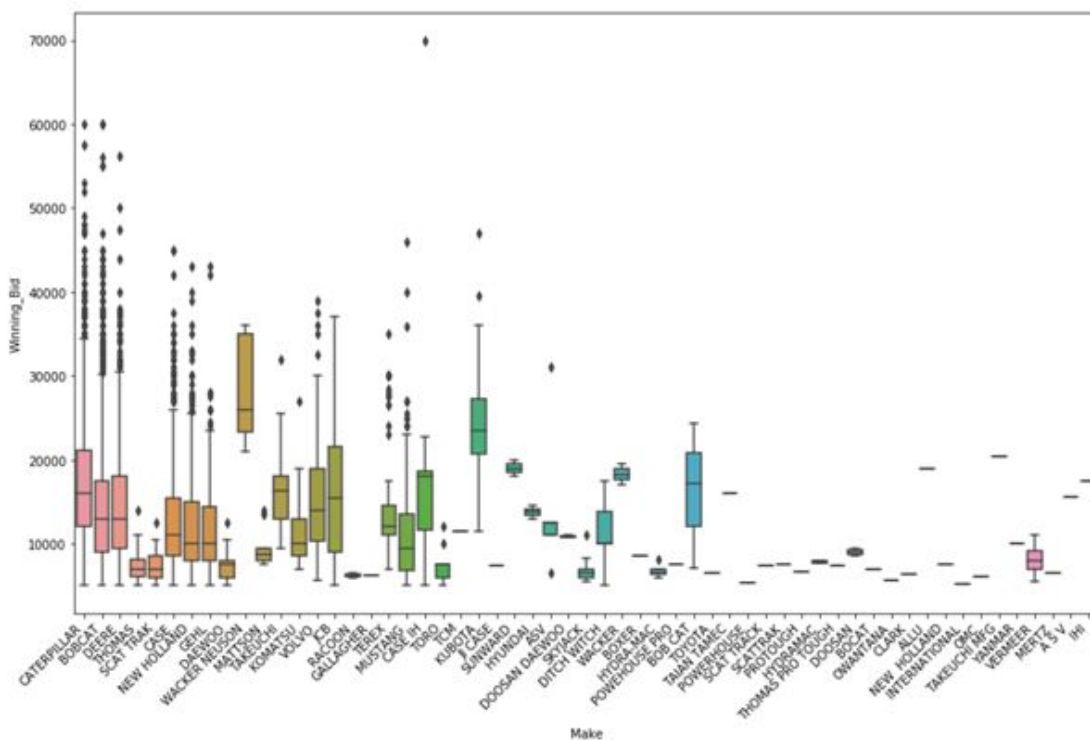


Figure 13. Boxplot of Winning Bid vs. Make of the Skid Steer

Boxes represent data (winning bids of different skid steers of a particular make) within 1st and 3rd quartiles while black points are outliers.

From *Figure 13*, we can see the impact of brand premium to winning bid. We found that brands such as "Wacker Neuson" have a high average winning bid, but it has a small amount of data points compared with common brands such as "Caterpillar", "Bobcat" and "Deere". Common brands have high variance, which means their prices depend more on the different conditions of skid steers. Other brands on the right side of the plot, such as "New holland" don't have a lot of data points.

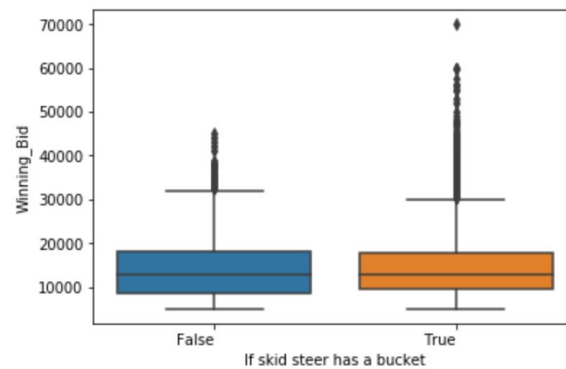


Figure 14. Winning Bid vs. Bucket

We treated "Bucket" as a binary predictor and made an assumption that labelled data had a bucket whereas null data didn't. According to the observation in the missing data section, this is not a 100% valid assumption. We'll be using MTurk to get more accurate data on add-ons. Also, this is the only column that can be treated as a categorical variable with minimal cleansing. Because it has the fewest missing values among all other categories: tires, transmission, etc and other categories can't be treated as a with/without problem. From *Figure 14*, we can see that the variance of "true" and "false" categories are almost the same, but "true" has more high winning bid outliers. We'll explore the possibilities of treating "Bucket" as a multi-category variable in the future.

- **Text data**

"Engine", "Tires" and "Transmission" suffer from the same problem. They all have more than 20 major categories whereas almost half of the labels only appear once. In addition, their columns are really sparse. We take "Engine" as an example. When you rank the subcategory of engine in descending order (*Figure 15 A*), you can see some major categories. We took a step further to see labels with more than 100 characters (*Figure 15 B*). They are essentially detailed descriptions of the state of the engine. Comments with green underline are positive while red underline represents negative. An alternative way to deal with text data is using text analysis to extract sensitive words like "damage", "new", "missing", "changed", etc and give them a positive or negative value. We will try to take care of some interesting cases such as skid steer with great age of sale but "damaged engine" for instance.

Ultra Low Sulfur Diesel	177
Four cylinder diesel engine	160
Kubota four cylinder diesel engine	54
Four cylinder turbo diesel engine	46
Case four cylinder diesel engine	29
Kubota four cylinder turbo diesel engine	15
Caterpillar four cylinder turbo diesel engine	14
Caterpillar four cylinder diesel engine	14
4 Cylinder Diesel Engine	13
Deutz four cylinder diesel engine	12

Figure 15. A. Major categories of 'Engine'

3294 46 hp, four cylinder diesel engine, <u>new factory remanufactured head, new pins and bushings in the bobtach, changed engine oil and filters, new battery</u>				
3682 engine access door latch area damaged, engine knocks, engine shroud damaged, new holland 450nc 5.0l four cylinder diesel engine				
3907 81 hp, approximately one hour on replaced engine, new holland 445m2 four cylinder diesel engine, <u>recently replaced engine</u>				
4075 70 hp, john deere 5030ht014 3.0l five cylinder turbo diesel engine, one engine cover side panel missing				
4148 <u>engine compartment fire damage</u> , john deere pe5030 four cylinder turbo diesel engine, lift arm cylinder pins removed, but included				
4269 bobcat 2.4l four cylinder diesel engine, check engine light on , <u>engine is not operating at full power</u>				

Figure 15. B. Sample of 'Engine' comments (more than 100 characters)

2) Other Interesting Feature Engineering Approaches

- **Glass Detection**

From the windshield's transparency/defect extent, intuitively, we can also tell degree of use of a skid steer. However, prior research focused on detecting breaches, dents, abrasions of products on the assembly line. Researchers used essentially enlarged, fine-grained images with little or no confounding factors (for instance, background), which are very different from our problem. In addition, glass detection needs a separate model. Using two models without high true positive rates to detect glass defect is not a feasible way in our opinion. Therefore, we moved on to "rust detection", which has more established research results.

- **Corrosion Detection through Image Processing**

Skid steer usually has a bucket attached to its front, which is used for moving dirt around. Different skid steers have buckets with different conditions, such as different levels of rust (corrosion) and various amounts of dirt covered on the exterior. This feature, however, is not reflected in the tabular data, but thought to be important in affecting the final price.

Due to the Nature of lacking labeled images in our data, we first think about using image processing techniques to extract corrosion features. Different researchers have done several experiments on this topic. Diaz ^[6] introduced an image processing pipeline, including thresholding (identify red color pixels above some threshold), edge detection (identify textures of rust) and segmentation algorithms (calculate the amount of rust) that could output a binary outcome of whether this image contains rust or not. Acosta ^[7] proposed another image processing pipeline for detecting rust zones. They utilized Perlin Noise to simulate an extreme rusted version of the input rust image. Then they feed both of the simulated and real images to

a filtering and feature extractor. Finally, they use the Bayes decision function to classify images into a binary output, rust or no rust.

These two approaches share the same pros and cons. They only require a small amount of labeled data (20 labeled images in Diaz's experiment) to reach a reasonably good result. Also, the thresholding (or filtering) and edge detection (or feature extractor) are highly tunable and interpretable. However, they all produced a binary output, which might not be enough for our use case. Most of the auction equipment are pre-owned and they usually have rust on them, so these models might produce imbalanced feature scores. Also most of the images used in these models contain little or no surroundings that are irrelevant to rust prediction, but we have various background objects in our images, making it harder to correctly capture rust and ignore the noise from the background. Last but not least, from a practical point of view, these methods do not have Python implementation, so we need extra coding efforts to implement them.

- **Corrosion Detection through Transfer Learning**

To compensate for the issue of lacking labeled images, we also considered using transfer learning. The idea is to use a small amount of labeled image to adapt a pre-trained model to our task of corrosion detection.

The choice of model depends on the final task, and few papers talked about this. As for the training architect and structure of transfer learning, Yosinski (2014)^[8] set up several experiments for comparison. They quantitatively proved that transfer learning of images classification works great for pretrained models (features extractor) on new data. Especially, a fine-tuned transfer learning model, even if only fine-tuned the last layer, works almost always better than a frozen transferred model, as long as transferring to a newly unseen dataset. However, there is also an accuracy degradation when transferring to a new target task. In this case, fine-tuning more layers could help alleviate this effect. This paper offers a general guidance of training transfer learning models.

We used a pre-trained VGG16 model as our feature extractor. Although it was not the most accurate image classification model for now, it has a simple structure and is one of the most widely used one for transfer learning. We changed the dense layer to match our task and fine-tuned all the dense layers and the last convolution layer (Conv 5-3) (see *Figure 16*). We also transformed images to match the input assumptions of the original VGG16 model and augmented images using flipping, rotating and color jittering to account for different shooting angles and rust color.

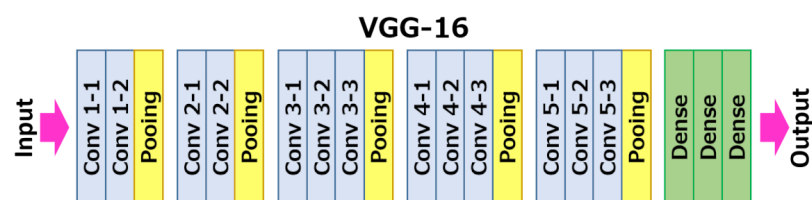


Figure 16. Structure of VGG16 model

We performed transfer learning on 2 different annotated dataset, 100 images with 5-point-scale of rust and 218 images with 3-point-scale of rust. The best accuracy on test set is 32% and 43% respectively.

3) Model Benchmarks

We benchmarked the performance of different models or different hyperparameter sets. Except for the currently testing hyperparameter, we control every other hyperparameter the same. By default we use a pretrained ResNet152 model, trained on 70% of all the preprocessed data and validated on the rest 30%. We train for 50 epochs, with learning rate starting with 0.001 and diminishing by 0.1 for every 10 epochs. We use mean square error as the loss function and Stochastic Gradient Descent (SGD) as the optimizer. We only use one fully-connected layer to predict price.

- **Comparing Different Pre-trained Models**

Transfer learning requires us to choose a pre-trained model and adapt it to our use case. This table shows the performance comparison between different image feature models. Other hyperparameters are controlled to be the default value. We finally choose to use ResNet152 as our pretrained

Pretrained Models	Loss (MSE)	Mean Absolute Error	Training Time
VGG19_bn	0.055372	3466.98	2.38
ResNeXt50	0.038298	2840.69	1.72
ResNeXt101	0.034485	2674.65	4.32
ResNet18	0.045419	3120.65	1.00 (39m 58s)
ResNet50	0.036007	2710.06	1.40
ResNet152	0.034755	2697.24	2.78

Table 6: Performance comparison between various pre-trained models. Loss is the mean square error of the price in the log-scale. Training time shows the relative time needed for training each model, and the absolute time 39m 58s may differ among different hardwares.

- **Comparing Different Blocks for Fine-tuning**

Fine-tuning a block means that the values of the parameters (weights) in this block will be trained (optimized) to fit into our use case. The pretrained ResNet152 model contains 4 convolutional blocks (CB). We benchmarked the performance for fine-tuning different numbers

of these blocks. Other hyperparameters are controlled to be the default value. Our findings coincide with Lathuilière (2018)^[9], showing that only fine-tuning the last 2 convolutional blocks of the pre-trained ResNet model would have the best performance.

Fine-tuning CBs	Loss (MSE)	Mean Absolute Error
All	0.034755	2697.24
CB3+	0.034420	2681.50
CB4+	0.036643	2769.05

Table 7: Performance comparison between various numbers of blocks for fine-tuning. CB4+ means only fine-tuning the last convolutional block and CB3+ means fine-tuning the last two convolutional blocks for the pretrained ResNet152 model.

• Comparing Different Fully-connected Layer Structures

Our goal is to predict price, which is a numerical value with only 1 dimension. We use different sets of fully-connected layers to transform previous network information, with 2053 dimensions, to 1 dimensional price. By default we only use one fully-connected layer to predict price, but this is not complex enough to the price prediction task. Thus we explore the possibilities of using multiple fully-connected layers with various numbers of hidden nodes. We also try adding a dropout layer (DO) to avoid overfitting. We control all other hyperparameters to be the default values, except for only fine-tuning the last 2 convolutional blocks (CB3+). Results show that adding more complexity could greatly improve the performance, and using only 32 hidden nodes will be the best one.

Fully-connected Structure	Loss (MSE)	Mean Absolute Error
(2053, 1)	0.035242	2682.30
(2053, 32, 1)	0.032363	2570.16
(2053, 64, 1)	0.032585	2585.95
(2053, 256, 1)	0.032884	2612.20
(2053, 32, DO, 1)	0.035349	2683.15
(2053, 64, DO, 1)	0.034404	2661.82
(2053, 256, DO, 1)	0.033620	2653.15
(2053, 256, 32, 1)	0.033229	2615.56

(2053, 256, 64, 1)	0.033109	2631.03
--------------------	----------	---------

Table 8: Performance comparison among different fully-connected layer structures. (2053, 1) is the baseline model where only one fully-connected layer is used. (2053, 32, DO, 1) means two fully-connected layers are used and with 32 hidden nodes and a dropout layer in between.

- **Comparing Different Optimizers and Loss Functions**

Optimizers are algorithms used to optimize model parameters (weights) during training. Loss functions are the target value that optimizers optimize on. We benchmarked the performance of using different optimizers among SGD, Adam and Adad and different loss functions among Mean Square Error (MSE), Mean Absolute Error (MAE) and Huber loss (HUB). Other hyperparameters are controlled to be the default value. Results show that using SGD optimizer and MSE loss function will have the best marginal performances.

Optimizers	Loss (MSE)	Mean Absolute Error
SGD	0.035359	2690.22
Adam	0.048680	3180.59
Adad	0.050003	3230.64

Table 9: Performance comparison between different optimizers.

Loss Functions	Loss (MSE)	Mean Absolute Error
MSE	0.035359	2690.22
MAE	0.151675	2753.34
HUB	0.018265	2736.53

Table 10: Performance comparison between different loss functions. Mean Square Error (MSE)

- **Comparing Different Activation Functions**

Activation functions are the non-linearity parts in neural networks. We benchmarked the performance of using different activation functions in the fully-connected layers. We test sigmoid function (sig) and hyperbolic tangent function (tanh). Other hyperparameters are controlled to be the default value. Results show that hyperbolic tangent function outperforms sigmoid with a small margin.

Activation Functions	Loss (MSE)	Mean Absolute Error
Tanh	0.032363	2570.16
Sigmoid	0.033578	2657.56

Table 11: Performance comparison between Sigmoid and Tanh activation functions.

4) MTURK Survey Design

4a: Questionnaire Design

1. Does this equipment have a bucket?

a. If “yes”: Use a scale of 0-10 to describe how much rust it has.

Click on “View instructions” for example of different scale of rust.

b. If “no”: jump to the second question.

2. Please look at the background, is it a sunny weather? (Yes/No)

3. Please look at the background, is there blue sky in the image? (Yes/No)

4. Please look at the background, is there green pasture or grass on the ground? (Yes/No)

5. Please look at the background, is it a cloudy weather? (Yes/No)

6. Please look at the background, is it a snowy(foggy) weather? (Yes/No)

7. Please look at the background, is the image taken outdoor? (Yes/No)

8. Use a scale of 0-10 to describe the conditions of equipment’s brand label. 0 means the poorest conditions (scratchiness, faded color, etc.) and 100 means great as new.

Click on “View instructions” for example of different scale of scratchiness.

(Slider value)

9. Use a scale of 0-10 to describe the degree of tire situation. 0 means totally worn out and 100 means totally new tire.

Click on “View instructions” for example of different degree of tire wear.

(Slider Value)

10. Use a scale of 0-10 to describe the brightness of paint. 0 means totally faded and 100 means bright as new.

We'll provide standard color of yellow (F0B823), white and green in the survey.

(seems like can only get an approximate of sunbelt green and bobcat white)



(Slider Value)

11. Use a scale of 0-10 to describe the brightness/blackness of black cabin frame on the top of vehicle. 0 means totally faded and 100 means bright as new.

12. Use a scale of 0-10 to describe how clean is the skid steer in general. 0 means fully covered with dirt and 10 means totally clean as new.

Click on "View instructions" for example of different dirt levels.

4b: Detailed instructions for workers to label bucket rust extent.

Summary	Detailed Instructions	Examples
<p>We'll provide several examples of bucket with different extent of rust.</p> <p>1. No Rust</p> <div></div> <p>Left-hand-side image: There is no reddish coating on the bucket. The bucket has its original color black.</p> <p>Right-hand-side image: There is dirt in the bucket but no obvious rust appeared.</p>		

2. Rust with scale 1



Slight rust on the front of the bucket. Not a lot.

3. Rust with scale 5



Medium amount of rust on the bucket. Rust is not all over the bucket.

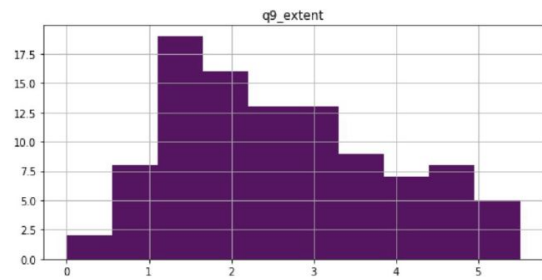
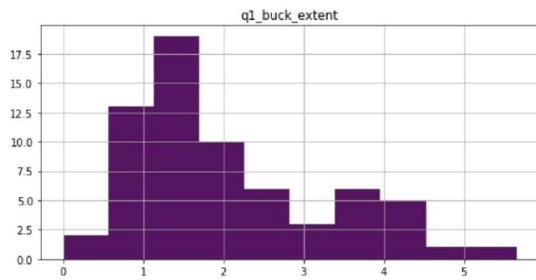
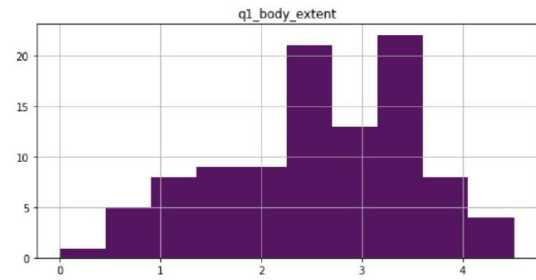
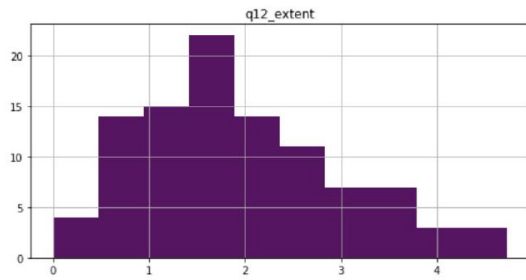
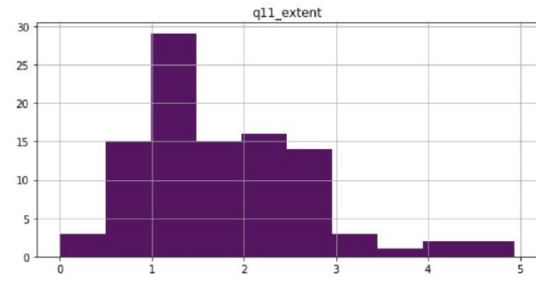
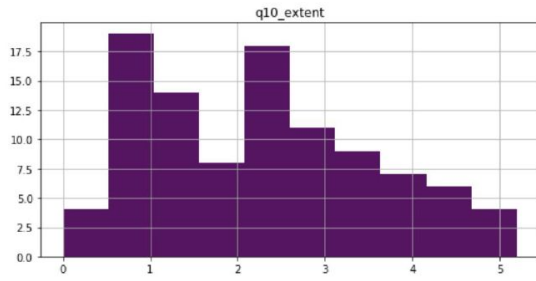
4. Rust with scale 10



Rust is all over the bucket.

Figure 17: Detailed instructions for workers to label bucket rust extent.

4c: MTURK Annotation Quality



Distribution of Standard Deviations of All Numerical Answers for Each Image

Graph 18: Question numbers correspond to the questionnaire design in 4a. Each standard deviation is calculated from 3 different answers per question per image, and the distribution is calculated upon the 100 images per question. The middle graph in the right column centers more towards the right side, meaning that people have more divergent opinions about this question (the amount of rust on the body of skid steer).

5) Categorical Feature Importance Breakdown:

1) Month_of_sold_date

OLS Regression Results						
=====						
Dep. Variable:	winning_bid		R-squared:	0.401		
Model:	OLS		Adj. R-squared:	0.399		
Method:	Least Squares		F-statistic:	237.1		
Date:	Sat, 21 Mar 2020		Prob (F-statistic):	0.00		
Time:	02:28:35		Log-Likelihood:	-5417.8		
No. Observations:	4625		AIC:	1.086e+04		
Df Residuals:	4611		BIC:	1.095e+04		
Df Model:	13					
Covariance Type:	nonrobust					
=====						
	coef	std err	t	P> t	[0.025	0.975]

hours_final	-0.0302	0.012	-2.489	0.013	-0.054	-0.006
age_at_sale	-0.6220	0.012	-52.159	0.000	-0.645	-0.599
Apr	0.0784	0.043	1.804	0.071	-0.007	0.164
Aug	-0.0338	0.048	-0.705	0.481	-0.128	0.060
Dec	-0.0071	0.035	-0.205	0.838	-0.075	0.061
Feb	-0.0616	0.039	-1.574	0.115	-0.138	0.015
Jan	-0.1456	0.084	-1.740	0.082	-0.310	0.018
Jul	0.0644	0.047	1.368	0.171	-0.028	0.157
Jun	-0.0239	0.034	-0.703	0.482	-0.091	0.043
Mar	0.0184	0.030	0.606	0.544	-0.041	0.078
May	0.0300	0.044	0.690	0.490	-0.055	0.115
Nov	-0.0808	0.045	-1.797	0.072	-0.169	0.007
Oct	0.2720	0.041	6.574	0.000	0.191	0.353
Sep	-0.0957	0.033	-2.931	0.003	-0.160	-0.032

Figure 19: Results of Multiple Linear Regression on (Y: winning_bid vs X: hours_final, age_at_sale, month_sold)

Based on the model's coefficients with month's p-value lower than 0.05 being chosen, we conclude that October and April have a positive influence on price while January and September have a negative influence on price.

2) Make

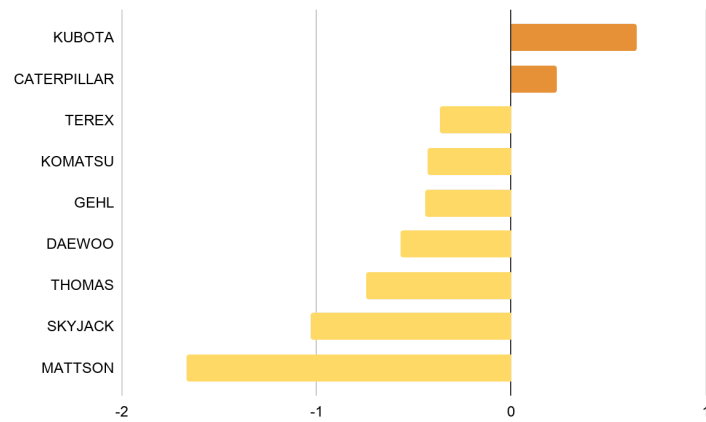


Figure 20: Brand's influence towards price prediction.

Color orange represents positive influence (brand's premium) while color yellow represents negative influence. This chart is based on the following regression model's coefficients with brands p -value lower than 0.05 being chosen.

	coef	std err	t	P> t	[0.025	0.975]
hours_final	-0.0292	0.012	-2.512	0.012	-0.052	-0.006
age_at_sale	-0.6016	0.012	-50.022	0.000	-0.625	-0.578
ALLU	0.8551	0.762	1.123	0.262	-0.638	2.348
ASV	-0.1661	0.539	-0.308	0.758	-1.222	0.890
BOBCAT	0.0016	0.017	0.094	0.925	-0.032	0.035
CASE	0.0305	0.032	0.953	0.341	-0.032	0.093
CATERPILLAR	0.2357	0.026	8.999	0.000	0.184	0.287
CLARK	-1.5462	0.762	-2.030	0.042	-3.040	-0.053
DAEWOO	-0.5697	0.254	-2.242	0.025	-1.068	-0.071
DEERE	-0.0106	0.036	-0.293	0.770	-0.081	0.060
DITCH WITCH	-1.1768	0.762	-1.545	0.123	-2.670	0.317
DOOSAN	-0.6249	0.539	-1.160	0.246	-1.681	0.431
GALLAGHER	-2.932e-15	5.2e-15	-0.563	0.573	-1.31e-14	7.27e-15
GEHL	-0.4390	0.076	-5.763	0.000	-0.588	-0.290
HYDRAMAC	-0.9008	0.341	-2.644	0.008	-1.569	-0.233
HYUNDAI	-0.0123	0.762	-0.016	0.987	-1.506	1.481
IHI	-7.661e-17	2.54e-15	-0.030	0.976	-5.06e-15	4.91e-15
INTERNATIONAL	-1.9636	0.762	-2.578	0.010	-3.457	-0.470
JCB	0.2227	0.131	1.704	0.088	-0.034	0.479
KOMATSU	-0.4262	0.180	-2.374	0.018	-0.778	-0.074
KUBOTA	0.6432	0.147	4.366	0.000	0.354	0.932
MATTSON	-1.6648	0.242	-6.892	0.000	-2.138	-1.191
MUSTANG	-0.3512	0.087	-4.017	0.000	-0.523	-0.180
NEW HOLLAND	-0.1379	0.045	-3.051	0.002	-0.227	-0.049
OMC	-1.6267	0.762	-2.136	0.033	-3.120	-0.133
OWANTANA	-2.203e-15	7.45e-16	-2.959	0.003	-3.66e-15	-7.44e-16
POWERHOUSE	-2.2376	0.762	-2.937	0.003	-3.731	-0.744
PROTOUGH	-1.4044	0.762	-1.844	0.065	-2.898	0.089
RACCOON	-1.6877	0.762	-2.215	0.027	-3.181	-0.194
SCAT TRAK	-0.1367	0.242	-0.566	0.571	-0.610	0.337
SKYJACK	-1.0268	0.220	-4.668	0.000	-1.458	-0.596
TAIAN TAMEC	-0.4427	0.762	-0.581	0.561	-1.936	1.051
TAKEUCHI	-0.0796	0.197	-0.404	0.686	-0.466	0.307
TEREX	-0.3656	0.101	-3.616	0.000	-0.564	-0.167
THOMAS	-0.7438	0.185	-4.020	0.000	-1.107	-0.381
TORO	-1.0879	0.381	-2.856	0.004	-1.835	-0.341
TOYOTA	0.0182	0.762	0.024	0.981	-1.476	1.513
VERMEER	-1.2305	0.440	-2.798	0.005	-2.093	-0.368
VOLVO	-0.2852	0.106	-2.697	0.007	-0.493	-0.078
WACKER NEUSON	0.5357	0.381	1.406	0.160	-0.211	1.283
YANMAR	0	0	nan	nan	0	0

Figure 21: Results of Multiple Linear Regression on (Y: winning_bid vs X: hours_final, age_at_sale, Make)